Data Verification in Integrated RFID Systems

Sakai, Kazuya; Sun, Min-Te; Ku, Wei-Shinn; Lu, Hua; Lai, Ten-Hwang

Published in:
I E E E Systems Journal

DOI (link to publication from Publisher):
10.1109/JSYST.2018.2865571

Publication date:
2019

Document Version
Accepted author manuscript, peer reviewed version

Data Verification in Integrated RFID Systems

Kazuya Sakai, *Member, IEEE*, Min-Te Sun, *Member, IEEE*, Wei-Shinn Ku, *Senior Member, IEEE*, Hua Lu, *Senior Member, IEEE*, and Ten H. Lai

***Abstract*—Radio frequency identification (RFID) is widely used as a tagging system to facilitate physical transactions in the real world. Thanks to the availability of inexpensive passive RF tags, RFID technology is now the catalyst of Internet of Things, i.e., every object can be uniquely identified in an Internet-like way. In the future, many individual RFID systems are likely to be integrated into a few exascale RFID systems. In an integrated RFID system, service providers (SPs) that offer RFID-based data service and clients that use the data service are different organizations. As a consequence, quality of data in terms of authenticity is of significant concern. In this paper, we first formulate a data verification problem in RFID systems and build a model of integrated RFID systems where multiple SPs and clients exist. Then, we propose two data verification protocols to ensure data generated by SPs associated with a particular tag and has not been modified. In addition, we implement our system as a prototype. The computer simulations, analyses, and testbeds based on the prototype all demonstrate that the proposed verifiable integrated RFID system achieves a high level of security and performance.**

***Index Terms*—Data security, radio frequency identification, RFID tags.**

I. Introduction

RADIO frequency identification (RFID) has emerged as an electronic tagging technology, where RF tags are used as the unique identifier of objects. Its wide adoption significantly reduces the cost of inventory management and facilitates a number of transactions in the physical world, such as library management [1], indoor localization [2], [3], warehouse operations [4], and so on. In addition, RFID technologies serve on the catalyst of the Internet of Things (IoTs), where a unique ID is assigned to every object in the physical world. The key to the success of RFID technology is the availability of inexpensive passive RF tags. Although passive tags do not have a power source, they can be energized by signals from RF readers and are capable of simple computations, e.g., 16-bit pseudorandom generator, a collision resistant hash function, etc.

While RFID drives a number of personal and business applications, security and privacy threats are always a concern for individuals and organizations. Hence, many studies have been devoted to an RF reader securely obtaining tag IDs by private authentications [5]–[7], jamming-based private authentications [8], [9], secure grouping protocols [10], [11], and to verifying an owner's credential by a motion signature [12] or tag activation [13]. A securely obtained tag ID is used as a pointer to the data entry in the back-end server. However, to the best of our knowledge, there is no study on the authenticity of data. Therefore, we are interested in the data verification problem in RFID systems.

In RFID systems, the back-end database server stores the information about objects or information generated based on objects' status. Thus, any piece of data is associated with a particular tag. A set of data $D_T = \{d_1, d_2, \ldots, d_i\}$ associated with Tag T is said to be *verifiable* if it can be proved that D_T is the information about the object referred by T and any element of D_T cannot be modified without being detected by the owner of T. Therefore, we first formulate the formal definition of the data verification problem in RFID systems as follows: *A challenger provides data set D_T associated with Tag T, and a verifier can verify that all elements in D_T are associated with T and none of them are modified.*

One of the applications is an integrated RFID system. At present, different RFID systems use different tagging systems. However, individual RFID systems may converge into only a few single tagging systems in the near future. In an integrated RFID system, RFID technology is not only an identification system, but also the source of valuable information. In other words, an RFID system generates a huge amount of sensitive data by reading tags.

The advantages of integrated RFID systems are as follows.

1) Integrating multiple RFID systems into an exascale system reduces the operational cost and hardware cost for tag-deploying organizations. At present, it is common for an item to have multiple tags attached, each from a different organization, and each of these organizations keeps an entry in their own database for the item. If these RFID systems are integrated, only a single tag and a single database will be required to track an item.
2) An integrated RFID system realizes a variant of the real-name system, where any user/object must register an

Manuscript received December 3, 2017; revised April 11, 2018 and June 29, 2018; accepted August 7, 2018. This work was supported in part by the Ministry of Science and Technology under Grants MOST107-2218-E-011-012, MOST107-2221-E-008-082-MY2, and MOST107-2218-E-001-006 and in part by the National Science Foundation under Grants IIS-1618669 (III) and ACI-1642133 (CICI). *(Corresponding author: Min-Te Sun.)*

K. Sakai is with the Department of Electrical Engineering and Computer Science, Tokyo Metropolitan University, Tokyo 191-0065, Japan (e-mail: ksakai@tmu.ac.jp).

M.-T. Sun is with the Department of Computer Science and Information Engineering, National Central University, Taoyuan 320, Taiwan (e-mail: msun@csie.ncu.edu.tw).

W.-S. Ku is with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: weishinn@auburn.edu).

H. Lu is with the Department of Computer Science, Aalborg University, DK-9220, Aalborg East, Denmark (e-mail: luhua@cs.aau.dk).

T. H. Lai is with the Department of Computer Science and Engineering, Ohio State University, Columbus, OH 43210 USA (e-mail: lai.1@osu.edu).

Digital Object Identifier 10.1109/JSYST.2018.2865571

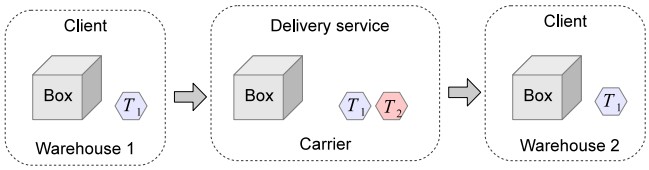


Fig. 1. Example of a tagging system, where multiple tags are attached to an item.

account on a blog, website, or social service with her/its real name. For instance, Facebook [14] does not allow one person to have multiple accounts under different e-mails. Having every object in the world associate with a single tag could provide identification credential of objects to RFID-based data services.

For example, the warehouse as well as the carrier company manages their shipments via RF tags. Consider the scenario that a warehouse ships a box to a customer or another branch via a ground transportation service. Two tagging systems are involved as shown in Fig. 1. One is the RFID-based inventory management system in the warehouse labeled by T_1; the other is the system employed by the carrier, which attaches a tag to each box for delivery services labeled by T_2. In an integrated RFID system, only one tagging system exists. In other words, a single tag for a box is shared by the inventory system and the carrier.

Although integrating multiple RFID systems into an exascale system has advantages, the authenticity of data is of concern, since users and providers of RFID-based data may belong to different organizations. Thus, the data verification problem must be addressed. To construct a verifiable integrated RFID system, in this paper, we propose two data verification protocols to ensure the authenticity of the data, which are generated by semitrusted organizations. Note that semitrusted model is generally used in cloud-based services [15], [16]. The proposed integrated RFID system is similar to cloud-based services in the sense that data are generated and maintained by service providers (SPs), and these SPs follow the prescribed protocol and do not collude. To validate the performance and the level of security of the proposed verifiable integrated RFID system, numerical analyses and computer simulations have been conducted. To demonstrate the feasibility of the proposed verifiable integrated RFID system, a prototype as well as testbeds based on the prototype have been built. Specifically, the contributions of this paper are as follows.

1) We define the data verification problem in RFID systems.
2) We model an integrated RFID system architecture, where the ownership of RF tags remains but other organizations can read these tags and generate valuable information.
3) We propose a data verification protocol, called 1-1 protocol, for the verifiable RFID system to ensure the authenticity of data generated by SPs.
4) We generalize the proposed data verification protocol into the m-n protocol, where m clients and n SPs exist. The proposed general model is practical in terms of key storage cost and computational cost in each party.
5) We implement a prototype of the m-n protocol, and complete testbeds to demonstrate the feasibility of the verifiable RFID system.

The rest of this paper is organized as follows. Related works are studied in Section II. In Section III, verifiable integrated RFID system architecture is introduced. We propose data verification protocols for integrated RFID systems in Section IV. In Section V, we conduct valid data rate analyses and computational cost analyses of the proposed data verification protocols. The performance of the proposed system is evaluated by computer simulations in Section VI and by testbeds in Section VII. In Section VIII, we provide our conclusion and suggest a few possible future directions of this research.

II. Related Work

The problem of data verification in integrated RFID systems is related to verifiable database systems. In general, for a query from a client, the server provides data and its proof (i.e., the authenticator of the data). A database is said to be *verifiable* if a client can check that his/her data in the untrustworthy database server is correct in the sense that any other party cannot add/delete/modify his/her data without being detected. As authenticated data structures, Merkle tree [17], distributed Merkle tree [18], one-way accumulators [19], skip-lists [20], and hash tables [21] are widely used. For example, in the tree-based approach, data records in the database are mapped to leaf nodes and each node maintains the authenticator for a data record.

Requirements of data verification are different from application to application. In some database systems, data records should be stored in a nonerasable and nonrewritable format to establish the irrefutable proof and accurate details of past events [22]. Li *et al.* [23] proposed a Merkle hash tree-based data retention and verification mechanism with write-once and read-many properties in rewritable storage media. In their tree structure, the authenticator of the root is directory updated without the authenticator of internal nodes when a leaf node is updated due to data addition.

On the other hand, in cloud computing environments, it is natural for a client to update data in a server. Banabbas *et al.* [24] developed a verifiable computation scheme that allows a client to efficiently update data and its proof in the database server. In verifiable data streaming [25], the order of streamed data (e.g., a client streams data to a storage server) is considered, and data verification is guaranteed in the database with an unbounded size.

The data verification problem in RFID-based databases is somewhat similar to verifiable database systems, but different due to the following reasons. First, an integrated RFID system has write-once and read-many properties. For clarification, the write-once and read-many property is applied to the data entry in database, but the tag's memory is rewritable. Second, data are generated by reading tags, and the amount of data in the database can increase exponentially. Third, the order of data generated by an RFID system is of concern. In addition, the data verification in RFID systems differs from general verifiable database systems in the definition of verifiability. Existing

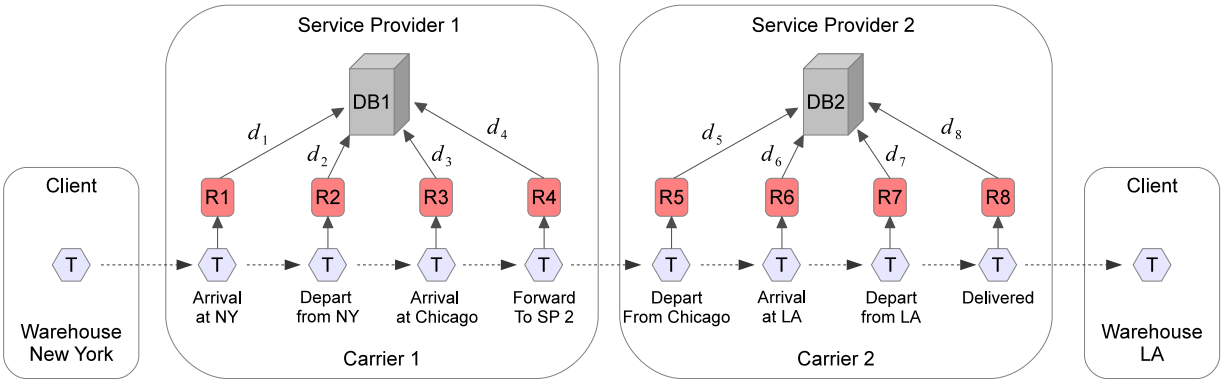


Fig. 2. System model of the proposed integrated RFID system.

verifiable database systems guarantee that the data entry in the database is not modified without the permission of its client and the index of the data entries being correct. In this research, the verifiability is defined in the sense that each data entry in the database is associated with a particular tag.

III. Problem Formulation

A. Integrated RFID Architecture

As we discussed in the aforementioned warehouse example, different RFID systems use different tag population at present. Thus, a single object may have multiple tags during its life cycle labeled by different organizations, i.e., a warehouse and a carrier. Since objects with a tag are uniquely identifiable (i.e., the idea of the IoTs), in the near future, individual RFID systems may converge into a single integrated exascale RFID system. In the aforementioned example shown in Fig. 2, the warehouse has the ownership of the box and the tag, and each carrier generates data by reading the tag during the delivery service. In other words, a single tagging system exists between two different RFID applications. This proposed architecture significantly reduces the operational and hardware cost for tag deployment.

In our verifiable RFID systems, two kinds of parties exist, *clients* and *SPs*. A client is an organization that owns objects with an RF tag, such as supermarkets and warehouses. A client has its own RFID systems for inventory management purposes. During supply chain, its products are processed and managed by other organizations, called SPs. An SP is a semitrusted organization that reads RF tags, generates data, and stores tags' information in its database. Thus, the organization that owns objects with a tag is a client with respect to the organization that provides information service by reading these tags and generating data. The system architecture is illustrated in Fig. 2. In this figure, there are two SPs (Service Provider 1 and Service Provider 2). Each SP has its database, denoted by DB, and a number of readers, denoted by R. Tag T owned by the client is read by R_i $(1 \leq i \leq 8)$ during the delivery process of these SPs, and data d_i is generated in every tag access.

An organization could be either a client or an SP for a particular tag population. In addition, many organizations are involved in an integrated RFID system. Thus, for particular tag populations, we define the m-client and n-SP model, where each client has a number of tags and they are processed by SPs.

B. Data Verification Problem

The data verification is a process to ensure the authenticity of data. Particularly, in RFID systems, the data stored in the back-end server is the information about objects or information generated based on objects' status. Thus, any piece of data is associated with a particular tag. In a traditional RFID system, tags are read and generated data is used by the same organization. An organization can always ensure that data generated by reading tags are associated with a particular tag. Thus, the authenticity of generated data has not been of concern. However, when it comes to integrated RFID systems, clients and SPs are different organizations. Therefore, the clients must be able to ensure the authenticity of data provided by SPs.

The authenticity of data is defined as the verifiability. A set of data $D_T = \{d_1, d_2, \ldots, d_i\}$ associated with Tag T is said to be *verifiable*, if we can prove that D_T is the information about the object referred by T and any element of D_T cannot be modified without being detected. An integrated RFID system is said to be *verifiable* if a client can verify the authenticity of any data set D_T generated by SPs, where T is any tag that the client owns.

C. Assumptions

In addition to passive RF tag functions defined by EPC Global Gen 2 [26], tags are assumed to be able to execute the *synchronization* command. That is, a tag is capable of computing a hash value of a key and updating its key, i.e., $Key \leftarrow H(Key)$, where Key is a key and $H(.)$ is a collision resistant hash function. The synchronization technique is used in many studies [27] to prevent adversaries from tracking a tag. Thus, the tag's memory is assumed to be rewritable, which is a necessary condition to provide security and privacy mechanisms in RFID systems.

An SP is semitrusted in the sense that the SP does not physically compromise tags. For example, as defined by EPC Global Gen 2 [26], a tag has unreadable memory space by readers, where access and kill passwords are stored. We assume an SP neither changes the password of a tag nor kills a tag by physical attacks. In addition, an SP is assumed not to send false data to tags. Let us consider that a client, say a warehouse, ships a box

TABLE I
DEFINITION OF THE NOTATIONS USED IN THIS PAPER

Symbols	Definition
V_i	The client i
T_i	The tag T_i
SP_i	The service provider SP_i
TK	Tag's secret key
RK	SP's secret key
D_T	Data set associated with Tag T
C	A counter
N_r	The random number generated by a reader
N_t	The random number generated by a tag
$H(.)$	A hash function
π	The signature of a reader
σ	The signature of a tag

with a tag to a customer via a semitrusted ground transport SP. Should the SP physically compromise or kill the tag, it will be penalized by the law or hurt the credibility of the organization. Hence, there is no motivation for the SP to do such things. However, the SP could generate data without reading tags or modify information in its database. For example, it is common for multiple carriers, each with an RFID system, to be involved in a single shipment as illustrated in Fig. 2. When an item is delivered using express mail, the carriers are given a strict schedule to follow so that the item can be shipped to the customer on time. In case of a shipment delay, the customer usually can get his/her money back from the carrier responsible for the delay. If no verification mechanism is provided, an SP (i.e., the carrier who is responsible for the delay of the shipment) can change the generated data of the tags in the database, then the customer has no way of finding who is responsible for the delay.

The replies from tags may collide during interrogations, which can be handled by collision avoidance mechanisms [28], [29]. This paper focuses on the data verification protocol, in which how to write data to tags and how to manage these data are discussed. Therefore, we assume that the reader eventually can access individual tags during the read/write process.

IV. DATA VERIFICATION PROTOCOL

In this section, we propose a data verification protocol for verifiable integrated RFID systems. The notations used in this paper are listed in Table I.

A. Overview of Data Verification Protocol

The data verification is achieved by exchanging signatures between an SP and a tag. The proposed verification protocol consists of three phases. The first phase is system initialization, where the client generates two keys, a tag's key TK and a reader's key RK. TK is assigned to a tag, and RK is assigned to an SP. In addition, the client provides a counter $C = 0$ to the SP. The second phase is data generation. In this stage, the SP reads tags and generates data d. Based on RK, C, and d, the SP computes a signature for data d, and by the query-and-response, the tag also computes a signature and replies to the SP. In the third phase, the client verifies the authenticity of the data generated by the SP by the signature of the SP and tag.

First, we introduce the 1–1 protocol as a baseline for a simplified integrated RFID system, where one client and one SP exist. Then, we will propose a practical data verification protocol for the m-client and n-SP model for arbitrary m and n values.

B. 1–1 Protocol

In the 1-client and 1-SP model, one client and one SP exist. The 1–1 protocol for the data verification in this simplified model consists of three phases, which is elaborated in the following sections.

1) System Initialization: The client initializes the system by assigning a key TK for tags and RK for the SP. In addition, the client provides a counter C with initial value 0 to the SP. The tag's secret key TK is stored in unreadable memory space in the tag, and thus the SP cannot obtain TK from a tag unless it physically compromises the tag.

2) Data Generation: When SP accesses T, an RF reader is involved in the communication. For simplicity, we just say SP sends a query to T, T replies to SP, and so on. In the data generation phase, both SP and T generate a signature for the data verification. RK and TK are used to compute a signature, and for each interrogation, both RK and TK are updated by the $synch$ command. Note that in the literature [27], the $synchronization$ command is used to update the common secret between a reader and a tag, hence the name is $synch$. RK_i and TK_i are computed by applying a hash function i times, and the bases are $RK_0 = RK$ and $TK_0 = TK$.

For each interrogation, SP reads Tag T and generates d_i, which is the ith data associated with T. SP chooses a random number $N_{r,i}$ and computes a signature π_i for d_i. Note that the use of random numbers prevents the replay attack, in which an adversary clones a tag's reply seen before. π_i is obtained by a hash function $H(RK_i, d_i||C||N_{r,i})$, where $||$ represents the concatenation of two binary strings. Then, SP sends π_i and $N_{r,i}$ to T. On receiving a query, T also generates a random number $N_{t,i}$ and computes a signature σ_i by $H(TK_i, ID||\pi_i||N_{r,i})$, where ID is T's identifier. Then, T sends σ_i and $N_{t,i}$ to SP. On receiving T's replay, SP stores data d_i and the proof (π_i, σ_i, C, $N_{r,i}$, $N_{t,i}$) to the database. SP updates the key by $RK_{i+1} \leftarrow H(RK_i)$, and increments the counter by 1, i.e., $C \leftarrow C + 1$. Finally, SP sends the $synch$ command to T. With the $synch$ command, T computes $TK_{i+1} \leftarrow H(TK_i)$ and stores TK_{i+1} in the memory. Note that the old key is overwritten and replaced by the new key. The pseudocode is given in Algorithm 1.

3) Data Verification: In the data verification phase, Client V obtains data D_T from SP and verifies the data authenticity in terms that all data in D_T are associated with Tag T. Should SP modify any data or add data without reading T, V is able to detect.

First note that V knows the RK, TK, and ID of T. V requests the ith data of T, and SP replies with d_i and its proof (π_i, σ_i, C, $N_{r,i}$, $N_{t,i}$). Based on the counter C, V computes the keys RK'_i and TK'_i by applying $H(RK)$ and $H(TK)$ i times, respectively. With these keys, V computes two signatures π'_i by

Algorithm 1: Data generation phase.

SP		Tag
generates data d_i generates a random number $N_{r,i}$ $\pi_i \leftarrow H(RK_i, d_i\|\|C\|\|N_{r,i})$	$\xrightarrow{\pi_i, N_{r,i}}$	
		generates a random number $N_{t,i}$ $\sigma_i \leftarrow H(TK_i, ID\|\|\pi_i\|\|N_{t,i})$
	$\xleftarrow{\sigma_i, N_{t,i}}$	
stores data and the signatures d_i, $(\pi_i, \sigma_i, C, N_{r,i}, N_{t,i})$		
	$\xrightarrow{synch}$	
updates the secret key and the counter $RK_{i+1} \leftarrow H(RK_i)$ and $C \leftarrow C+1$		updates the secret key $TK_{i+1} \leftarrow H(TK_i)$

Algorithm 2: Data verification phase.

Client		SP
requests the i-th data of Tag T	$\xrightarrow{T,i}$	
		finds d_i of Tag T
	$\xleftarrow{d_i,(\pi_i,\sigma_i,C,N_{r,i},N_{t,i})}$	
computes the keys and signatures $RK'_i \leftarrow \{H(RK)\}^C$ $TK'_i \leftarrow \{H(TK)\}^C$ $\pi'_i \leftarrow H(RK'_i, d_i\|\|C\|\|N_{r,i})$ $\sigma' \leftarrow H(TK'_i, ID\|\|\pi_i\|\|N_{t,i})$ if $\pi'_i = \pi_i$ and $\sigma'_i = \sigma_i$, then d_i is valid otherwise, d_i is invalid		

$H(RK'_i, d_i||C||N_{r,i})$ and σ'_i by $H(TK'_i, ID||\pi_i||N_{r,i})$. Then, V checks whether π'_i equals π_i and σ'_i equals σ_i. If so, d_i is valid. Otherwise, d_i is invalid. The pseudocode is provided in Algorithm 2.

C. m-n Protocol

In this section, we propose the m-n protocol for data verification in the m-client and n-SP model. Let V_i be Client i, and SP_j be SP j. Assume each client V_i owns l_{V_i} tags, and these tags could be processed by all SPs. The straightforward approach based on the 1–1 protocol requires the key storage cost of $n \times l_{V_i}$ for clients, n for tags, and $\sum_{i=0}^{m} l_{V_i}$ for SPs. This is because the key and counter are different for each SP. Thus, this approach is impossible, since tags can store only a few keys due to the storage constraint. For instance, in EPC Global Gen 2, 32-bit keys are used, and a tag normally has less than 512 bits memory space.

Hence, we propose the m-n data verification protocol with the key storage costs of $n + l_{V_i}$ for clients, 1 for tags, and m for SPs, respectively. The proposed protocol consists of three phases, system initialization, data generation, and data verification.

1) System Initialization: Let $RK_{j,i}$ be the key assigned by a client V to compute the signature for the ith data $d_{j,i}$ generated by SP_j. For data $d_{j,i}$ associated with T owned by V, Client V must be able to compute the corresponding TK_k and $RK_{j,i}$ from TK and RK. Note that the number of synchronization commands applied to $RK_{j,i}$ and TK_k is different, since a number of SPs may read Tag T. Hence, in addition to SPs, each tag T needs to keep a counter C_T.

Each client, say V_i, generates $RK_{j,0}$ for each SP SP_j, and TK_0 for each tag T. In addition, V initializes the counter C_{SP_j} to be 0 for each SP_j and the counter C_T for each T. Thus, Client V_i stores $n + l_{V_i}$ keys, SP SP_j stores m keys and m counters, and Tag T of any client stores one key and one counter.

2) Data Generation: In the m-n protocol, Tag T owned by Client V will be processed by a number of SPs, say SP_j. The counter C_T and C_{SP_j} for all SPs that generated data from T are incremented. Thus, we have $C_T = \sum_{\forall SP_j} C_{SP_j}$ as long as all the SP obeys the protocol.

Let $D_{j,T} = \{d_{j,1}, d_{j,2}, \ldots, d_{j,i}\}$ be the data set generated by SP_j and associated with T. Similar to the 1-client and 1-SP model, SP_j generates the ith data $d_{j,i}$, generates a random number, and computes a signature $\pi_{j,i}$ by using $RK_{j,i}$, C_{SP_j}, and $N_{r,i}$. Then, SP_j sends the signature and the random number to Tag T. When the tag creates a signature, it incorporates its counter C_T. The counter value will be $C_T = k$, where $k = \sum_j C_{SP_j}$ for all SP_j that reads T so far. On receiving the signature σ_i, the counter C_T, and $N_{t,i}$, SP_j stores data d_i and its proof ($\pi_{j,i}$, σ_k, C_{SP_j}, C_T, $N_{r,i}$, $N_{t,i}$). Finally, SP_j and T update their key and counter. The pseudocode is provided in Algorithm 3.

3) Data Verification: In the m-n protocol, Client V obtains the ith data $d_{j,i}$ and its signature from SP_j. Since each SP

Algorithm 3: Data generation phase.

Service Provider SP_j		Tag T
generates data $d_{j,i}$		
generates a random number $N_{r,i}$		
$\pi_{j,i} \leftarrow H(RK_{j,i}, d_{j,i}\|\|C_{SP_j}\|\|N_{r,i})$		
	$\xrightarrow{\pi_{j,i}, N_{r,i}}$	
		generates a random number $N_{t,k}$
		$\sigma_k \leftarrow H(TK_k, ID\|\|\pi_{j,i}\|\|C_T\|\|N_{t,k})$
	$\xleftarrow{\sigma_i, N_{t,k}}$	
stores data and the signatures		
$d_{j,i}$, $(\pi_{j,i}, \sigma_k, C_{SP_j}, C_T, N_{r,i}, N_{t,k})$		
	$\xrightarrow{synch}$	
updates the secret key and the counter		updates the secret key
$RK_{i+1} \leftarrow H(RK_i)$ and $C_{SP_j} \leftarrow C_{SP_j} + 1$		$TK_{k+1} \leftarrow H(TK_k)$ and $C_T \leftarrow C_T + 1$

Algorithm 4: Data verification phase.

Client V		Service Provider SP_j
requests the i-th data of Tag T from SP_j		
	$\xrightarrow{T,i}$	
		finds $d_{j,i}$ of Tag T
	$\xleftarrow{d_{j,i},(\pi_{j,i},\sigma_k,C_{SP_j},C_T,N_{r,i},N_{t,k})}$	
computes the keys and signatures		
$RK'_{j,i} \leftarrow \{H(RK)\}^{C_{SP_j}}$		
$TK'_k \leftarrow \{H(TK)\}^{C_T}$		
$\pi'_{j,i} \leftarrow H(RK'_{j,i}, d_{j,i}\|\|C_{SP_j}\|\|N_{r,i})$		
$\sigma'_k \leftarrow H(TK'_k, ID\|\|\pi_{j,i}\|\|C_T\|\|N_{t,k})$		
if $\pi'_{j,i} = \pi_{j,i}$ and $\sigma'_k = \sigma_k$, then $d_{j,i}$ is valid		
otherwise, d_i is invalid		

updates its secret key and counter independently, even if one of the SPs adds or modifies data without reading a tag, other SPs are intact.

In the data verification protocol, V first requests the ith data associated with T to SP_j, and then SP_j returns $d_{j,i}$ and ($\pi_{j,i}$, σ_i, C_{SP_j}, C_T,$N_{r,i}$, $N_{t,k}$). V computes the corresponding keys $RK'_{j,i}$ for $\pi'_{j,i}$ and TK_k for σ_k by applying the hash function C_{SP_j} and C_T times, respectively. Here, k is the number of reads by SPs, i.e., $k = \sum_{\forall\ SP_j} C_{SP_j}$. If $\pi'_{j,i} = \pi_{j,i}$ and $\sigma'_k = \sigma_k$, then the data $d_{j,i}$ is valid. Otherwise, it is invalid. The pseudocode is provided in Algorithm 4.

D. Optimization

To verify the authenticity of data, a client must compute a number of hash functions, which may take a long time. Let N_d be the number of data generated from a tag. A client can request the ith data generated by the jth SP, and $ij <= N_d$ always holds. Without the key caching, $i \times j$ computations are required for each data verification. To save the computational cost, we propose an optimization mechanism by means of the key caching. Our key caching mechanism minimizes the computational cost with a bounded size of key caching.

Let S be a set of keys and $S_{\max}$ be the number of keys that will be stored at a client for each tag. The current cache size is denoted as $|S|$. If $|S| < S_{\max}$, a client simply stores the current key in the cache. When $|S| = S_{\max}$, the client needs to discard the current key after data verification or replace an existing key with the new one. Note that $|S| > S_{\max}$ should not happen because $S_{\max}$ is the bounded size of the cache.

Each key in S corresponds to the ith data generated by the jth SP in some ways. We define the distance between two keys, $d(key_1, key_2)$, as the number of computations to obtain key_2 from key_1 by a hash function $H(.)$. That is, key_2 is obtained by applying the hash function $d(key_1, key_2)$ times. If key_2 cannot be obtained from key_1, $d(key_1, key_2) = \infty$. Let X be the random variable defined as $d(s_k, key)$, where key is the current key and s_k is in S. Our goal is to minimize $\sum_{i=1}^{N_d} \frac{X}{i}$. This can be done by scanning all keys in the cache. Note that the cache size is considered as a constant, as the cache size is normally very small compared to a sampling population.

If two keys key_1 and key_2 are valid, $d(key_1, key_2)$ can simply be computed by $i_2 j_2 - i_1 j_1$, where key_1 is the i_1th data generated by the j_1th SP and key_2 is the i_2th data generated by the j_2th SP. Assume kth element in S ($1 \leq k \leq |S|$) is the key for i_kth data generated by j_kth SP. In the proposed optimization mechanism, we first add a new key to S. Since S contains $|S| + 1$ keys at this time, we will remove one of the keys in S so that $\sum_{i=1}^{N_d} \frac{X}{i}$ is minimized. To optimize the computational cost, we need to find the kth key ($1 \leq k \leq |S|$) such that $i_{k-1} j_{k-1} - i_{k+1} j_{k+1}$ is minimized, where

Algorithm 5: Key caching algorithm.

```
/* Client does following */
if (|S| < S_max) then
    add the current key to S.
else
    add the current key to S.
    find s_k in S such that
        min_{1≤k≤S_max} (i_{k-1}j_{k-1} − i_{k+1}j_{k+1}).
    remove s_k from S.
```

$i_0 j_0 = 0$ and $i_{|S|+1} j_{|S|+1} = N_d$. Note that $i_1 j_1$ may be $i_0 j_0$, and $i_k j_k$ may be $i_{|S|+1} j_{|S|+1}$. The pseudocode is provided in Algorithm 5.

V. Analyses

A. Valid Data Rate Analyses Against Illegal Data Access

The valid data rate is an indicator to show how well a verification protocol protects tags' data against potential malicious SPs. Let p_{sp} be the probability that an SP is malicious, and p_d be the probability that a malicious SP illegally generates data, i.e., the SP does not obey the verifiable tag access protocol when it generates data. We denote the average number of data generated for a tag as N_d and the number of SPs that process a tag as N_{sp}.

First, we analyze the valid data rate for the 1–1 protocol. An illegal data access indicates that a malicious SP does not obey the protocol, and this causes the counters kept in SP and a tag not to be synchronized. Thus, in the 1–1 protocol, once invalid data are added to the data set, the other SPs cannot generate a valid signature for data generation. Let X be the random variable that the kth SP illegally accesses a tag first, and Y be the minimum index of the invalid data generated by the kth SP. Since all of the jth SPs ($1 \le k \le X-1$) follow the protocol, all data generated by the kth SP is valid. In addition, up to the $(Y-1)$th data are valid, but the Yth data are invalid. Thus, the valid data rate can be formulated by

$$\frac{N_r E[X] + E[Y]}{N_d}. \tag{1}$$

For simplicity, $M_{sp} = \lfloor N_{sp} p_{sp} \rfloor$ and $M_d = \lfloor N_r p_d \rfloor$. The expected values of X and Y are computed by the following:

$$E[X] = \frac{1}{N_{sp}} \sum_{i=1}^{N_{sp}} \binom{M_{sp}}{1} \frac{i}{N_{sp}} \binom{M_{sp}}{M_{sp}-1} \times \left(\frac{N_{sp} - i - 1}{N_{sp}} \right)^{M_{SP}-1} \tag{2}$$

$$E[Y] = \frac{1}{N_d} \sum_{i=1}^{N_d} \binom{M_d}{1} \frac{i}{N_d} \binom{M_d}{M_d-1} \times \left(\frac{N_d - i - 1}{N_d} \right)^{M_d-1}. \tag{3}$$

Next, we analyze the valid data rate for the m-n protocol. In this protocol, even though malicious SPs illegally access a tag, data generated by other SPs are intact. Thus, the valid data rate is independent of the random variable X. We deduce (4) for the valid data rate of the data verification protocol:

$$1 - N_{sp} p_{sp} + \frac{E[Y]}{N_r} N_{sp} p_{sp}. \tag{4}$$

B. Analyses of Computational Cost

We build an analytical model of the number of executions of a hash function in a data verification protocol. Without a key caching mechanism, a client must compute the corresponding key for a tag and an SP from the current keys. To analyze the computational cost, a random data access is considered.

Let X_c be the random variable that represents the distance between a base key and the current key, and X_n be the random variable that represents the distance between a base key and the next key. Assuming both current key and next key are valid, the number of computations can be obtained by X_n when $X_n < X_c$ and $X_n - X_c$ when $X_n > X_c$. Since a client is assumed not to request the same data, $X_n = X_c$ should never happen. Thus, we can derive the computation cost in

$$\frac{(X_c - 1) X_n}{N_d} + \frac{(X_d - X_c)(X_n - X_c)}{N_d}. \tag{5}$$

Next, we analyze the computational cost with the key caching. In our caching mechanism, the index of keys in the cache is uniformly distributed. Thus, given the size of the key cache $S_{\max}$, each pair of the closest keys in the cache is distanced by approximately $N_d / S_{\max}$. Therefore, the expected number of hash function computations can be obtained by

$$\frac{1}{N_d / S_{\max}} \sum_{k=1}^{N_d / S_{\max}} k = \frac{N_d / S_{\max} - 1}{2}. \tag{6}$$

VI. Performance Evaluation

We conducted computer simulations to evaluate the performance of the proposed data verification protocols, the 1–1 protocol and m–n protocol, along with the tree-based protocols [23].

A. Simulation Configurations

The integrated RFID system consists of 10 to 100 clients and SPs. Each client has 4096 tags, and each tag is processed by multiple SPs. Each SP reads a tag and generates data 10 to 100 times. Around 10% to 90% of SPs are malicious and they add data without reading a tag with probability p. The value of p ranges from 0.1 to 0.9. As attack models, illegal data access and illegal data modification are considered. The illegal data addition, where a malicious SP adds data without reading a tag, causes the signatures for subsequent data to be invalid. In the illegal data modification attack, a malicious SP modifies existing data entry in the database. For each tag, ten SPs are randomly selected that read the tag and generate data during its life cycle. In total, 1000 system realizations are generated and the average is taken as simulation results. The simulation parameters are shown in Table II.

TABLE II
SIMULATION PARAMETERS OF THE PERFORMANCE EVALUATION

Parameters	Values
Number of clients	10 to 100
Number of SPs	10 to 100
Number of tags	4096
Number of data accesses	10 to 100
Percentage of malicious adversaries	10% to 90%
Illegal data access/modification rate p	0.1 to 0.9
Number of simulation experiments	1000

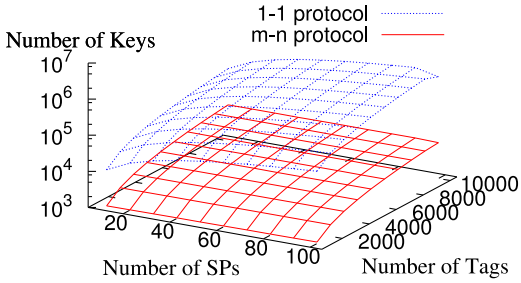


Fig. 3. Average key storage cost of each client for the different numbers of SPs and tags.

In this performance evaluation, the following metrics are considered.

1) *Valid Data Rate*—During the life cycle of a tag, SPs generate data including invalid data. A client randomly accesses data on an SP, and the proof for the data may or may not be valid. Valid data rate is defined as the number of data with valid proof divided by the number of data accesses.
2) *Number of Keys*—As key storage cost, the number of keys in the system is employed, including SP and tag keys that each organization (i.e., client or SP) maintains.
3) *Computational Cost*—A client has to compute the corresponding keys from the base RK and TK in the data verification phase. The number of hash functions applied to obtain the keys is used as computational cost.

B. Analytical Results

Fig. 3 demonstrates the key storage cost of each client with respect to the number of SPs. From analyses, it is clear that a client maintains $n + \overline{l_V}$ on average in the m–n protocol. Here, n is the number of SPs and $\overline{l_V}$ is the average number of tags that clients own. On the other hand, the 1–1 protocol incurs $n \times \overline{l_V}$ keys cost.

Fig. 4 shows the key storage cost of each SP with respect to the number of clients. Theoretically, an SP keeps $\sum_{i=0}^{m} l_{V_i}$ keys on average in the 1–1 protocol and m keys on average in the m–n protocol, where m is the number of clients. As shown in Figs. 3 and 4, the m–n protocol significantly reduces the key storage cost as indicated by the analyses.

Fig. 5 depicts the key storage cost with respect to the number of tags in the system. In this configuration, there are 100 clients and SPs. For the 1–1 protocol, each client and SP maintains the same number of keys, as the number of keys depends on the

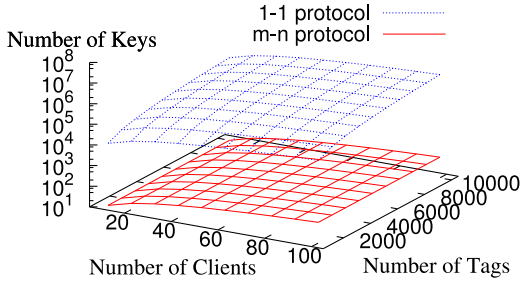


Fig. 4. Average key storage cost of each SP for the different numbers of clients and tags.

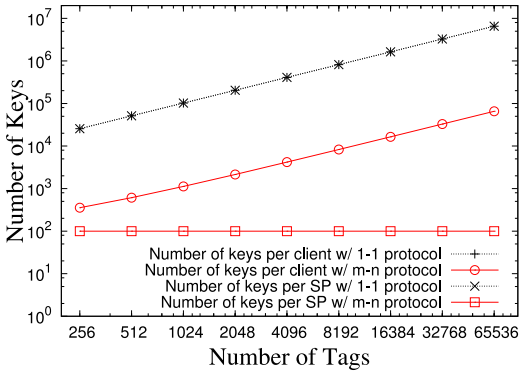


Fig. 5. Average key storage cost of an SP and a client for different number of tags.

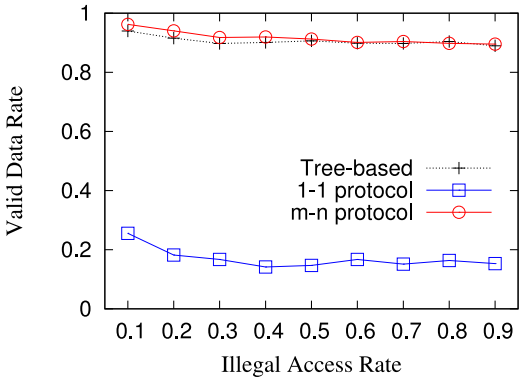


Fig. 6. Valid data rate for different illegal access rates.

total number of tags in the system. With the same reasons for Figs. 3 and 4, the storage cost per client (or per SP) with the m–n protocol is much smaller than that with the 1–1 protocol.

C. Simulation Results

Fig. 6 illustrates the valid data rate for the illegal data access attack with respect to illegal data access rate. As can be seen in the figure, even when the illegal access rate is 0.9, at least 90% of data has a valid proof in the m–n protocol and the tree-based protocol. This is because SP and tag independently update their key and counter. Hence, illegal data access affects signatures computed for data generated by one SP, and other data and its proof are intact. On the other hand, in the 1–1 protocol, once

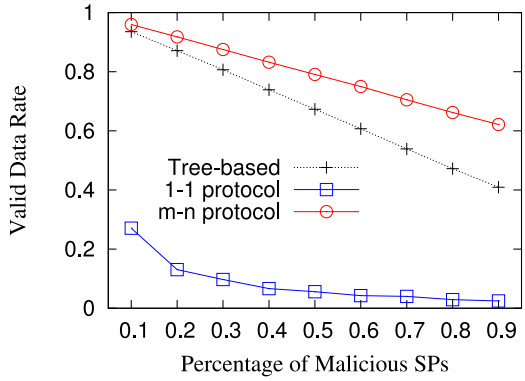


Fig. 7. Valid data rate under the illegal data access attack for different percentage of malicious SPs.

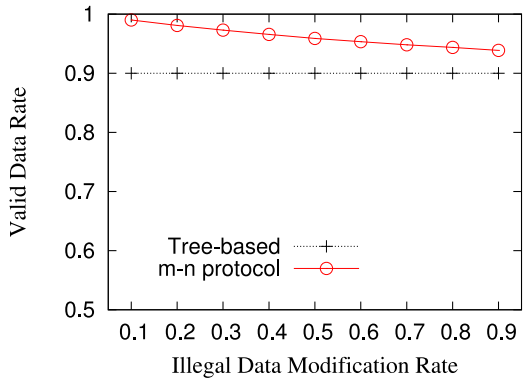


Fig. 8. Valid data rate for different data modification rates.

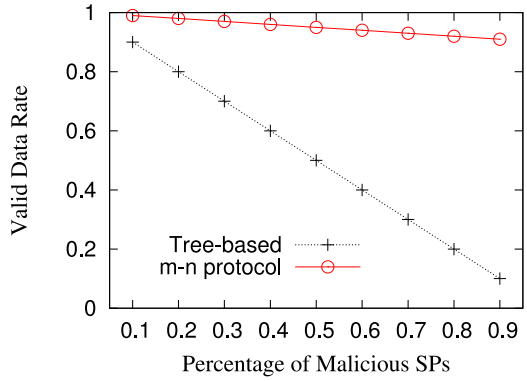


Fig. 9. Valid data rate under the data modification attack for different percentage of malicious SPs.

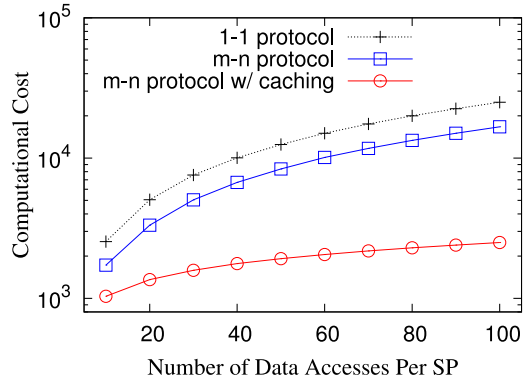


Fig. 10. Computational cost for different number of data accesses.

an SP adds data without reading a tag, all data generated after illegal access has invalid proof. As a result, the 1–1 protocol has poor valid data rate.

Fig. 7 presents the valid data rate for an illegal data access attack with respect to the percentage of malicious SPs. In this configuration, the percentage of malicious SPs ranges from 10% to 90%, and each malicious SP illegally adds data with a probability of 10%. In the tree-based protocol, once a malicious SP adds data without reading a tag, it updates the signature for all internal nodes in the key tree. This causes a client not to access any data located in the tree. Hence, the valid data rate in the tree-based protocol drastically decreases according to the percentage of malicious SPs. With the same reason as Fig. 6, the 1–1 protocol results in poor performance even when there are only a few malicious SPs. From Figs. 6 and 7, it is clear that the proposed m–n protocol is more reliable than other protocols.

Fig. 8 shows the valid data rate for the data modification attack with respect to the data modification rate. Note that the 1–1 protocol and the m–n protocol has the same performance against the data modification attack, and thus we omit the 1–1 protocol. In this setting, 10% of SPs are malicious and modify existing data for a tag. In the m–n protocol, only modified data are affected, and thus the valid data rate linearly decreases as the illegal data modification rate increases. On the other hand, in the tree-based protocol, all signatures are updated in the internal nodes in the tree should one of its data for a tag be modified, and hence a client cannot obtain a valid signature for any data entry for the tag in the malicious SP.

Fig. 9 presents the valid data rate for the data modification attack with respect to the percentage of malicious SPs. In this scenario, 10% to 90% of SPs are malicious and each malicious SP modifies 10% of data in its database. If data for a tag is modified in a malicious SP, a client cannot access any data for the tag. Thus, the valid data rate of the tree-based protocol decreases as the percentage of malicious SPs increases. On the other hand, our m–n protocol results in high valid data rate, since only modified data is affected and others are intact.

Fig. 10 shows the computational cost with respect to the number of data accesses for a particular tag. In this scenario, a client requests ith data from jth SP that processes the client's tag, where i and j are randomly selected. Without the key caching, the client must compute the corresponding keys from the base keys for each data verification, which causes heavy computational cost. By the key caching mechanism, the computational cost is alleviated by 90% as shown in Fig. 10.

Fig. 11 presents the computational cost with respect to the number of data generated by each SP. From this figure, we can see the key caching reduces 60% of computations compared with the m–n protocol. Though the computational cost linearly increases as the number of generated data increases, the key caching is still effective.

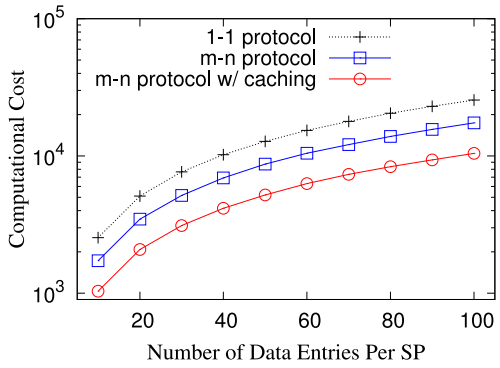


Fig. 11. Computational cost for different number of data entries.

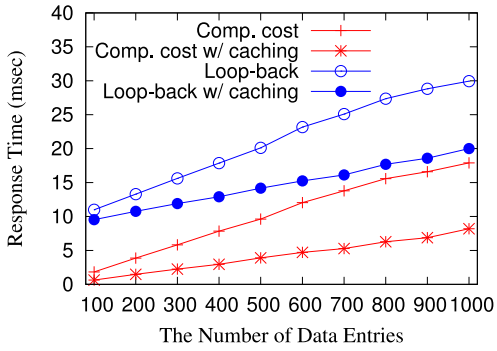


Fig. 13. Turn around time with the loop-back setting, where the client and server programs run in the same host.

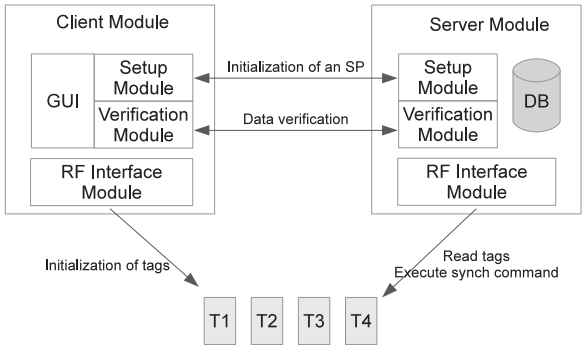


Fig. 12. Program modules of the client and server.

VII. Implementation and Testbeds

We have implemented a prototype of the m–n protocol to demonstrate the feasibility of our proposed data verification scheme in an integrated RFID system.

The prototype consists of a number of modules. The interactions between different modules are shown in Fig. 12. The system consists of the client side and server side modules. Both the client and server have setup, data verification, and RF interface modules. The setup module initializes an SP with the secret key and a counter. The data verification module is an implementation of Algorithm 4. The RF interface module is the program of an RF reader. At the client side, the RF interface module initializes the key and a counter of a tag. On the other hand, at the server side, RF interface reads a tag and executes the synchronization command. In addition, the client module has GUI.

A. Testbed Environment

The testbed is composed of two computers, since the verification process is conducted between a client and a server. One of the computers acts as the client; the other is the server (an SP). The data generated by an SP when it reads a tag is application dependent. Thus, in this testbed, the server module generates bulk information, such as "SP reads Tag 1 at 10:00 pm." In this testbed, we conducted two kinds of experiments as follows:

1) *Client-Server Tests*—In this testbeds, we will show that the proposed verification protocols are feasible in real network settings. To this end, three kinds of network configurations, the loop-back, LAN, and WAN accesses, are considered. In the loop-back configuration, both the client and server programs are executed in the same computer (MacBook Air), and the client program accesses the server program through the loop-back address, i.e., 127.0.0.1. The client (SP) computer acts as different clients (SPs) at different times, so that the m–n model is simulated. For the LAN setting, a Windows PC is used as a server in IEEE 802.11g wireless LAN controlled by a wireless broadband router (Linksys WRT54GL). In the WAN scenario, the server (Ubuntu PC) is located in National Central University, Taiwan, and the client accesses the server from The Ohio State University, Columbus, OH, USA. Experiments are conducted each hour in a day, and the average of each hour was computed. In all of the settings, MacBook Air is used as a client.
2) *Reader-Tag Tests*—In the proposed data verification system, read and write operations are performed during the data generation and data verification phases. In a read operation, an RF reader simply accesses a tag's content. On the contrary, in a write operation, a tag computes a hash function to synchronize the key and counter with an SP, as shown at the end of Algorithm 1 and 3. As an RFID system, we employ Motorola MC319Z [30] and passive tags.

Since the purpose of this testbed is to show the feasibility of our verifiable RFID system, the response time is considered. The response time is defined as the required time for a client to request data and verify its authenticity, and for an RF reader to complete a read/write operation. For each configuration, 100 experiments are performed, and the average values are collected as the results. Note that the system initialization is an offline process. Hence, we conducted several rounds of the testbed experiment to measure the performance of the online processes.

B. Client-Server Testbed Results

Fig. 13 illustrates the response time (ms) for the loop-back setting with respect to the number of data entries in the server. Since only one SP exists in this testbed, the number of data entries for a tag is set to be up to 1000. Comp. cost and Comp. cost w/ caching refer to the required time to compute the hash value from the base key, which does not contain the network-related delay. As shown in the figure, the response time increases as the number of data entries increases. This is because more

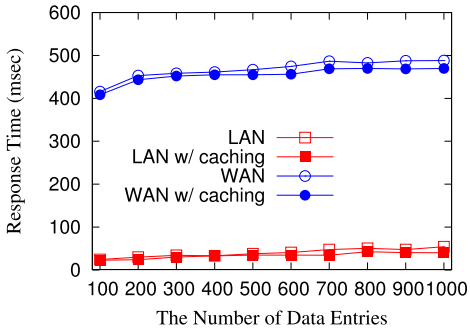


Fig. 14. Response time with LAN and WAN settings (in the WAN setting, the client host is located in Japan and the server is located in Taiwan).

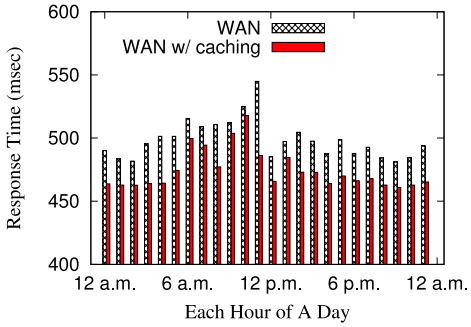


Fig. 15. Response time for different hours with the WAN setting.

data in the server implies that the client needs to apply the hash function more times to the base tag/SP keys for the proof. According to Fig. 13, the data verification process does not take much time. In addition, the key caching mechanism significantly reduces the response time.

Fig. 14 presents the response time (ms) for the LAN and WAN settings with respect to the number of data entries in the server. For both the LAN and WAN configurations, the key caching slightly reduces the response time, but the reduction is not significant. This implies that the response time with the data verification is mostly dominated by the network delay. Therefore, we can conclude that the computational delay introduced from the m–n protocol is very small, and the data verification in an integrated RFID system is feasible for real deployment in terms of the computational cost.

Fig. 15 shows the average response time (ms) for a WAN setting with respect to each hour of a day. The number of data entries is set to be 1000. The experiments had started at 12 a.m. on Mar 5, 2014 in the local time in the United States, and ended at 11 p.m. on the same day. As can be seen from the figure, the response times are different from time to time in a day. Particularly, the experiments at noon result in the slowest response time. However, the difference is not significant, since the response time is the order of milliseconds.

C. Reader-Tag Testbeds Result

Fig. 16 demonstrates the probability distribution function and cumulative distribution function of read and write operations time (ms). From the figure, we can say that most read and write operations are completed within 0.5 and 2.0 ms, respectively. On average, read operations take 0.384 s, and write operations take 1.46 s. Therefore, the proposed data verification protocol is feasible with a real RFID system.

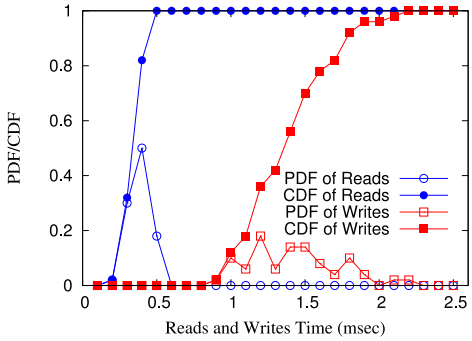


Fig. 16. Required time of read and write operations with Motorola MC319Z.

VIII. Conclusion

Security and privacy in RFID systems are some of the most significant concerns when we deploy RFID applications to the real world. While many RFID applications are deployed at present, those individual systems use different tagging systems. In the age of IoTs, individual RFID systems are merged into a few exascale RFID systems and use a single tagging system. In such a system, the authenticity of data must be addressed to improve the quality of RFID-based data service.

In this paper, we first propose an integrated RFID system, where a number of organizations are involved and a single tagging system exists. Then, we formulate the data verification problem, in which RFID-based data are verifiable in the sense that data are associated with a particular tag. To achieve this, we design two verification protocols, the 1–1 and the m–n protocols. To measure the degree of security and performance of our verifiable integrated RFID system, analytical models are built and computer simulations are conducted. In addition, we have implemented a prototype of the m–n model. From the testbeds for different network configurations, we conclude that the proposed verifiable RFID system is highly feasible.

For simplicity, we assume that each SP has one secret key. In the future, we plan to extend our work so that each SP is allowed to have multiple secret keys so that each branch of the SP has one secret key. In addition, there are other extended directions to this research. First, the scalability issue should be addressed when a large number of clients and SPs join the system. Second, the computational cost in the data generation and data verification processes could be further improved using a structured key management. Finally, we may extend our proposed system to limit the lifetime of tags' data based on the number of times tags are accessed.

References

[1] D. Molnar and D. Wagner, "Privacy and security in library RFID issues, practices, and architectures," in *Proc. 11th ACM Conf. Comput. Commun. Security*, 2004, pp. 210–219.

[2] S. Wagner, M. Handte, M. Zuniga, and P. J. Marron, "On optimal tag placement for indoor localization," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2012, pp. 162–170.

[3] J. Yu, W.-S. Ku, M.-T. Sun, and H. Lu, "An RFID and particle filter-based indoor spatial query evaluation system," in *Proc. 16th Int. Conf. Extending Database Technol.*, 2013, pp. 263–274.
[4] W.-S. Ku, H. Chen, H. Wang, and M.-T. Sun, "A Bayesian inference-based framework for RFID data cleansing," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2177–2191, Oct. 2013.
[5] F. Rahman, M. E. Hoque, and S. I. Ahamed, "Anonpri: A secure anonymous private authentication protocol for RFID systems," *Inf. Sci.*, vol. 379, pp. 195–210, 2017.
[6] M.-T. Sun, K. Sakai, W.-S. Ku, T. H. Lai, and A. V. Vasilakos, "Private and secure tag access for large-scale RFID systems," *IEEE Trans. Dependable Secur. Comput.*, vol. 13, no. 6, pp. 657–671, Nov./Dec. 2016.
[7] Y. Komori, K. Sakai, and S. Fukumoto, "Randomized skip graph-based authentication for large-scale RFID systems," in *Proc. Int. Conf. Wireless Algorithms, Syst., Appl.*, 2016, vol. 9798, pp. 1–12.
[8] K. Sakai, W.-S. Ku, R. Zimmermann, and M.-T. Sun, "Dynamic bit encoding for privacy protection against correlation attacks in RFID backward channel," *IEEE Trans. Comput.*, vol. 62, no. 1, pp. 112–123, Jan. 2013.
[9] K. Sakai, M.-T. Sun, W.-S. Ku, and T. H. Lai, "A novel coding scheme for secure communications in distributed RFID systems," *IEEE Trans. Comput.*, vol. 65, no. 2, pp. 409–421, Feb. 2016.
[10] A. Juels, "Yoking-proofs for RFID tags," in *Proc. 2nd IEEE Annu. Conf. Pervasive Comput. Commun. Workshops*, 2004, pp. 138–143.
[11] Y. Komori, K. Sakai, and S. Fukumoto, "RFID grouping protocol made private," in *Proc. IEEE Int. Conf. Dependable Syst. Netw.*, 2017, pp. 105–106.
[12] A. Czeskis, K. Koscher, J. R. Smith, and T. Kohno, "RFIDs and secret handshakes: Defending against ghost-and-leech attacks and unauthorized reads with context-aware communications," in *Proc. 15th ACM Conf. Comput. Commun. Security*, 2008, pp. 479–490.
[13] N. Saxena, M. B. Uddin, J. Voris, and N. Asokan, "Vibrate-to-unlock: Mobile phone assisted user authentication to multiple personal RFID tags," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2011, pp. 181–188.
[14] Facebook Inc. [Online]. Available: https://www.facebook.com. Accessed on: Aug. 24, 2018.
[15] H. Rong, H. Wang, J. Liu, and M. Xian, "Privacy-preserving k-nearest neighbor computation in multiple cloud environments," *IEEE Access*, vol. 4, pp. 9589–9603, 2016.
[16] W. Jiang, F. Li, D. Lin, and E. Bertino, "No one can track you: Randomized authentication in vehicular ad-hoc networks," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2017, pp. 197–206.
[17] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Proc. Annu. Int. Cryptol. Conf. Adv. Cryptol.*, 1988, pp. 369–378.
[18] M. T. Goodrich, J. Z. Sun, R. Tamassia, and N. Triandopoulos, "Reliable resource searching in peer-to-peer networks," in *Proc. Int. Conf. Security Privacy Commun. Syst.*, 2009, pp. 437–447.
[19] J. Camenisch and A. Lysyanskaya, "Dynamic accumulator and application to efficient revocation of anonymous credentials," in *Proc. 22nd Annu. Int. Cryptology Conf. Adv. Cryptol.*, 2002, pp. 61–76.
[20] M. T. Goodrich, C. Papamanthou, and R. Tamassia, "On the cost of persistence and authentication in skip lists," in *Proc. Int. Conf. Exp. Algorithms*, 2007, pp. 94–107.
[21] C. Papamanthou, R. Tamassia, and N. Triandopoulos, "Authenticated hash tables," in *Proc. Conf. Comput. Commun. Security*, 2007, pp. 437–448.
[22] R. Sion and M. Winslett, "Regulatory-compliant data management," in *Proc. Int. Conf. Very Large Databases*, 2007, pp. 1433–1434.
[23] T. Li, X. Ma, and N. Li, "WORM-SEAL: Trustworthy data retention and verification for regulatory compliance," in *Proc. Eur. Symp. Res. Comput. Security*, 2009, vol. 5789, pp. 472–488.
[24] S. Benabbas, R. Gennaro, and Y. Vahlis, "Verifiable delegation of computation over large datasets," in *Proc. Annu. Int. Cryptology Conf. Adv. Cryptol.*, 2011, pp. 111–131.
[25] D. Schroeder and H. Schroeder, "Verifiable data streaming," in *Proc. Conf. Comput. Commun. Security*, 2012, pp. 953–964.
[26] EPCglobal, "EPC radio-frequency identity protocols class-1 generation-2 UHF RFID protocol for communications at 860MHz-960MHz version 2.0.0," 2013.
[27] L. Lu, J. Han, L. Hu, Y. Liu, and L. M. Ni, "Dynamic key-updating: Privacy-preserving authentication for RFID systems," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2007, pp. 13–22.
[28] B.-H. Liu, N.-T. Nguyen, V.-T. Pham, and Y.-H. Yeh, "A maximum-weight-independent-set-based algorithm for reader-coverage collision avoidance arrangement in RFID networks," *IEEE Sensors J.*, vol. 16, no. 5, pp. 1342–1350, Mar. 2016.
[29] N.-T. Nguyen, B.-H. Liu, and V.-T. Pham, "A dynamic-range-based algorithm for reader-tag collision avoidance deployment in RFID networks," in *Proc. Int. Conf. Electron., Inf., Commun.*, 2016, 1–4.
[30] Motorola Solution Inc., [Online]. Available: http://www.motorolasolutions.com/.

Kazuya Sakai (S'09–M'14) received the Ph.D. degree in computer science and engineering from The Ohio State University, Columbus, OH, USA, in 2013.

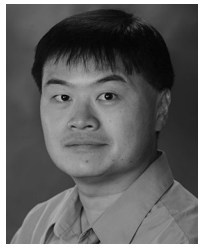
He is currently an Associate Professor with the Department of Electrical Engineering and Computer Science, Tokyo Metropolitan University, Hino, Japan. His research interests include the area of wireless and mobile computing, information and network security, and distributed algorithms.

Dr. Sakai received the IEEE Computer Society Japan Chapter Young Author Award 2016. He is a member of the ACM.

Min-Te Sun (S'99–M'02) received the B.Sc. degree from National Taiwan University, Taipei City, Taiwan, the M.Sc. degree from Indiana University Bloomington, Bloomington, IN, USA, and the Ph.D. degree in computer and information science from The Ohio State University, Columbus, OH, USA.

He is a Professor with the Department of Computer Science and Information Engineering, National Central University, Taoyuan City, Taiwan. His research interests include distributed computing and IoT.

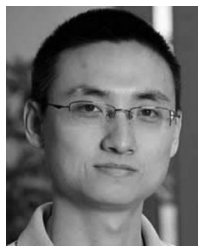
Dr. Sun is a member of the ACM.



Wei-Shinn Ku (S'02–M'07–SM'12) received the M.S. degree in computer science and in electrical engineering from the University of Southern California (USC), Los Angeles, CA, USA, in 2003 and 2006, respectively. He received the Ph.D. degree in computer science from USC, in 2007.

He is a Professor with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA. His current research interests include data management systems, data science, cybersecurity, and mobile computing. He has published more than 100 research papers in refereed international journals and conference proceedings.

Dr. Ku is a member of the ACM SIGSPATIAL.



Hua Lu (M'06–SM'14) received the B.Sc. and M.Sc. degrees from Peking University, Beijing, China, and the Ph.D. degree in computer science from National University of Singapore, Singapore.

He is an Associate Professor with the Department of Computer Science, Aalborg University, Aalborg, Denmark. His research interests mainly include database and data management, geographic information systems, and mobile computing.

Dr. Lu served on the editorial board of Mobile Information Systems from 2014 to 2016. He served as PC Co-chair or Vice Chair for ISA 2011, MUE 2011 and MDM 2012, demo chair for SSDBM 2014, and PhD forum Co-chair for MDM 2016. He has served on the program committees for many conferences such as VLDB, ICDE, CIKM, DASFAA, ACM SIGSPATIAL, SSTD, MDM, PAKDD, APWeb, and WAIM.

Ten H. Lai received the Ph.D. degree in computer science from the University of Minnesota, Minneapolis, MN, USA, in 1982.

He is currently a Professor of computer science and engineering at The Ohio State University, Columbus, OH, USA. He is interested in applying Zen to teaching and research.

Mr. Lai served as a program chair of ICPP 1998, general chair of ICPP 2000, program Co-chair of ICDCS 2004, general chair of ICDCS 2005, and recently, general Co-chair of ICPP 2007. He is/was an editor of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, *ACM/Springer Wireless Networks*, Academia Sinica's *Journal of Information Science and Engineering*, *International Journal of Sensor Networks*, and *International Journal of Ad Hoc and Ubiquitous Computing*.